Sidewinder: A Predictive Data Forwarding Protocol for Mobile Wireless Sensor Networks

Matthew Keally, Gang Zhou, Guoliang Xing†
Computer Science Department, College of William and Mary
†Department of Computer Science and Engineering, Michigan State University

***Abstract*—In-situ data collection for mobile wireless sensor network deployments has received little study, such as in the case of floating sensor networks for storm surge and innundation monitoring. We demonstrate through quantitative study that traditional approaches to routing in mobile environments do not work well due to volatile topology changes. Consequently, we propose Sidewinder, a predictive data forwarding protocol for mobile wireless sensor networks. Like a heat-seeking missile, data packets are guided towards a sink node with increasing accuracy as packets approach the sink. Different from conventional sensor network routing protocols, Sidewinder continuously predicts the current sink location based on distributed knowledge of sink mobility among nodes in a multi-hop routing process. Moreover, the continuous sink estimation is scaled and adjusted to perform with resource-constrained wireless sensors. Our design is implemented with nesC and evaluated in TOSSIM. The performance evaluation demonstrates that Sidewinder significantly outperforms state-of-the-art solutions in packet delivery ratio, time delay, and energy efficiency.**

I. INTRODUCTION

While most existing wireless sensor network deployments are still terrestrial networks with static sensor nodes, mobile wireless sensor networks have received increasing attention. During the past few years, several mobile wireless sensor networks have been successfully deployed in which sensor nodes are either equipped with motors for active mobility or attached to mobile objects for passive mobility. For example, researchers have attached wireless sensor devices to Micro Air Vehicles [1], bikes [2], vehicles [3] [4], and animals [5] [6]. In addition, wireless sensors are equipped with motors to move underwater to collect data from static sensor devices [7].

A general observation from these new deployments is that conventional routing solutions designed for static sensor networks have been discarded. For example, although tree-based routing protocols can tolerate minor topology changes in static deployments due to lossy links [8] [9], but they cannot survive excessive topology changes in mobile deployments [9]. Geographic forwarding-based routing protocols [10] [11] [12] [13] [14] are also discarded in these new deployments as intermediate nodes may not know the precise location of a mobile sink during a routing process.

Without choosing conventional solutions from static deployments, many mobile deployments [4] [3] [7] [2] employ delay-tolerant designs in which data collected by sensors is stored locally when connectivity is not available. The data is then opportunistically delivered to mobile sinks when connectivity is restored. While a delay-tolerant design serves the purpose of data collection for offline data analysis, it is not suitable for in-situ data collection in a more general mobile wireless sensor network application.

We now use an example of a buoyant sensor network for in-situ inundation monitoring to motivate the need for a holistic forwarding protocol for mobile wireless sensor networks. Since inundation prediction is largely based on modeling and simulation, it is imperative to collect runtime flooding information as in-situ feedback to simulation models. This prediction allows for correction of any simulation errors as well as adjusting simulation parameters for improved accuracy in following inundation predictions. A mobile wireless sensor network is an ideal platform for such in-situ data collection, in which sensors placed on buoys float with water currents. Hence, tracking individual sensors' locations gives the ability to compute the current inundation location, scope, moving speed, and direction. Sensors' location information can be obtained from GPS sensors and reported back to a few mobile sinks through multi-hop routing. A sink then communicates with onshore Internet access via long-range radio.

Before proposing our solution for a general mobile wireless sensor network that requires in-situ data collection, we first examine whether existing solutions for general mobile ad hoc networks can be directly used. Several classic routing protocols like [15] [16] [17] [18] have been developed for general mobile ad hoc networks. However, these solutions adopt the same strategy: establishing a routing path from the source to the sink before any data communication. Though this strategy works well with small topology changes due to node mobility, it does not apply with excessive topology changes.

More dynamic topology changes can be addressed using the techniques proposed in [19], where data is forwarded to an estimated area based on distance to the sink and mobility. However, routing decisions are only made independently at each hop using the information local to the current forwarding node. Most wireless sensor radios have a much smaller communication range (e.g. 10∼40m in MicaZ) than that in general mobile ad hoc networks (e.g. 150∼250m in 802.11). This smaller radio range generates more topology changes than in mobile ad hoc networks; a holistic estimation is needed to more accurately predict the sink location by aggregating information collected at each hop.

For these reasons, we propose Sidewinder, an in-situ data collection protocol for mobile sensor networks. Like a heat-seeking missile, Sidewinder data packets "home in" on the sink

via a combination of sink location prediction and estimation techniques. The main contributions of this work are:

- In wireless sensor network literature, there has been very little study on in-situ data collection with excessive topology changes due to sensor mobility. We perform a quantitative evaluation of traditional mobile ad hoc and wireless sensor routing protocols and determine that increased network dynamics cause the performance of these protocols to degrade significantly. The Sidewinder solution we propose is geared towards increased network dynamics due to node mobility and can handle cases of both random and group mobility.
- The Sequential Monte Carlo (SMC) theory is integrated into Sidewinder to handle intensive topology changes in mobile sensor networks. During a multi-hop data forwarding process, the distributed knowledge of a mobile sink location among intermediate nodes is integrated under SMC to make smart routing decisions. This aggregated approach contrasts with most traditional routing protocols in that Sidewinder continuously updates its routing path based on multiple sink location predictions.
- The SMC prediction approach in Sidewinder is scaled to reduce computation and bandwidth overhead, thus making it ideal for wireless sensor networks. Sink location estimates are clustered in such a way to minimize overhead while maintaining accuracy from hop to hop. This novel clustering technique surpasses other clustering and compression methods in minimizing computational complexity and communication overhead to allow sink location information to be piggybacked in data packets.
- Sidewinder is implemented with nesC [20] and evaluated in TOSSIM [21]. Extensive performance evaluation demonstrates that Sidewinder significantly outperforms existing solutions for in-situ data collection under intensive topology changes in mobile sensor netowrks.

This paper is organized as follows: Section II motivates this work with simulation and discussion of related work. Section III gives an overview of the Sidewinder design. In Section IV, we discuss the Sequential Monte Carlo Prediction-based data forwarding strategy. The performance evaluation is presented in Section V. In Section VI, we present conclusions.

II. Literature Study and Motivation

In general wireless ad hoc and sensor networks, a group of existing routing protocols perform a one-way [9] [22] or two-way [15] [16] [17] [18] [23] path discovery and use the discovered path for consecutive data communication. A sensor network has a much smaller radio range, typically $10 \sim 40$m, compared to a general wireless ad hoc network, $150 \sim 250$m. When excessive topology changes are observed in a sensor network, continuous maintenance of fixed routing paths becomes impractical for supporting effective communication. The approach in [19] provides a mobility-induced time and space adaptive beaconing mechanism to provide destination location information, but routing decisions are only made at each hop. We illustrate the detrimental effects of these topology changes on traditional wireless ad hoc protocols in Section II-A, which motivates the need for a protocol that makes routing decisions based on information accumulated at each hop to ensure data reaches the sink.

Another group of routing protocols [10] [11] [12] [13] [14] use periodic beaconing messages to discover neighbors, and use neighbors' geographic locations for local forwarding to achieve multi-hop communication. Some solutions exist to modify the neighbor table in the mobility case, such as [24] and [19]. When geographic locations are not available, some variants [25] [26] [27] use virtual coordinates or landmarks compared to selected anchor nodes for local forwarding. Like [19], [28] also provides a mobility-induced time and space adaptive beaconing mechanism to provide destination location information, but still makes use of a neighbor table. [29] [30] [31] use zone-based forwarding to address mobility, which still suffers in highly mobile environments. In Section II-B, we show through quantitative study how highly mobile environments cause traditional wireless sensor routing protocols to fail. This illustrates the need for a dynamic routing protocol with low overhead that does not rely on a neighbor table or fixed routes.

The results of the following quantitative studies illustrate new challenges that cannot be handled by traditional routing methods and motivate the design of our new routing protocol. In Section II-A, we show the impact of radio ranges on topology changes when nodes are mobile, concluding that traditional mobile ad hoc routing protocols do not work well for wireless sensors. In Section II-B, we show the detrimental performance impact of sink mobility on geographic forwarding-based protocols like GF [10] and its mobility-oriented enhancements [24].

A. Impact of Radio Ranges on Topology Changes

Taking AODV [15] as an example, we study how a routing protocol designed for general mobile wireless networks performs in a mobile wireless sensor network context. We implement AODV with nesC [20] in TOSSIM [21], and evaluate it with 500 nodes. A source-sink pair is chosen from the 500 nodes, and a multi-hop ($\sim$6 hops) path is established between them using AODV. We measure the path lifetime, which is defined as the time from which a path is established until it is broken, and study it under different node mobilities and radio ranges. For node mobility, we use the Random Waypoint [32] model without pause time [33]. We increase the radio range from 25m to 250m with 25m increments. To eliminate any effect of changing node density on end-to-end routing performance, we keep the node density at approximately 20 one-hop neighbors per node. To help focus on performance study of mobility, we do not simulate node failures or lossy links [8] [34] [9] so that their impact on routing performance can be separated. Two interesting observations are identified in Figure 1. First, the multi-hop end-to-end link has a much longer lifetime when the radio range is $150 \sim 250$m than when the radio range is $10 \sim 40$m. For example, the observed link lifetime goes up to 109s when the radio range is 250m,

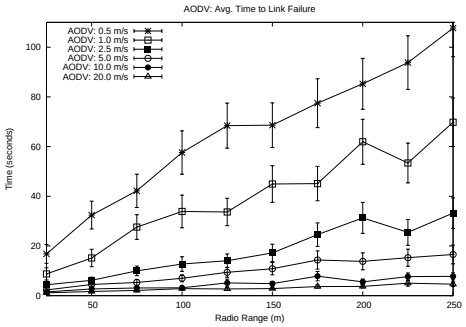


Fig. 1. Performance Impact of Radio Ranges on Topology Changes

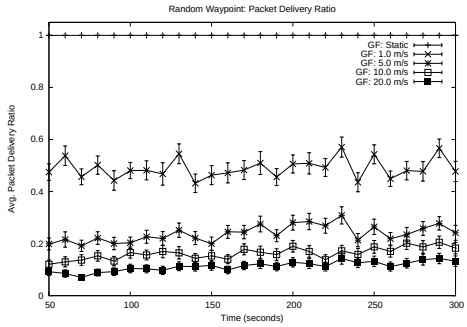


Fig. 2. Performance Impact of Sink Mobility on Geographic Forwarding

but it reduces to 18s when the radio range is 25m. This 84% link lifetime reduction indicates that even though AODV works well in general mobile wireless networks, it leads to very poor performance when it is applied to mobile wireless sensor networks. Second, when the node mobility is high, AODV leads to poor performance in both general mobile wireless and sensor networks. For instance, when the node mobility increases to 5+m/s, the path lifetime reduces to ≤ 11 seconds in general mobile wireless networks and ≤ 3 seconds in sensor networks. Routes with such a short lifetime are useless in practical systems, therefore, routing protocols that are originally designed for general mobile wireless networks [15] [16] [17] [18] can not be directly applied in mobile wireless sensor networks. Thus, a new protocol is required for wireless sensor networks that can handle high node mobility and accurately guide data from source to sink.

B. Impact of Sink Mobility on Geographic Forwarding

Geographic forwarding-based protocols [10] [11] [12] [13] [14] have been widely used in static wireless sensor networks, because they only maintain local information to achieve end-to-end routing. However, a common assumption of these geographic forwarding-based protocols is that all intermediate nodes in a routing path know the exact sink location and use it for multi-hop routing. This assumption is reasonable when the sink is static, but leads to poor performance when the sink is mobile. In this experiment, we use the same TOSSIM configuration as presented in the previous experiment to evaluate one of the most representative geographic forwarding-based protocols, GF [35], in mobile environments. GF is implemented with the Destination Location Prediction (DLP) mobility enhancement in [24]: when a forwarding node is a one-hop neighbor of the sink, the forwarding node selects the sink as the next hop. DLP is designed to eliminate local maximums, where a forwarding node is a one hop sink neighbor, but the forwarding node neighbor table indicates that it is the node closest to the sink location. As shown in Figure 2, when all nodes are static, the end-to-end packet delivery ratio is 100%. However, when nodes are mobile, the end-to-end packet delivery ratio drops sharply. For example, when the maximum node speed is 1m/s, which means a 0.5m/s average speed, the GF packet delivery ratio goes down to 50%. When the maximum node speeds increases further to 5m/s, 10m/s and 20m/s, the end-to-end packet delivery ratio reduces to $\leq 20\%$. There are two reasons why GF performs so poorly when nodes are mobile. First, the geographic forwarding direction is not continuously corrected based on distributed sink mobility knowledge among nodes in the routing path, so a source may fail to find a route to a mobile sink. Second, it is too costly to keep updating a neighbor table to deal with mobility, and an inadequate update usually leads to failure of data forwarding. Neighbor Location Prediction, presented in [24], predicts neighbor locations based on movements and attempts to reduce the beaconing frequency; zone-based forwarding described in [28] presents a more efficient alternative without the use of a neighbor table. While zone-based forwarding is integrated into our Sidewinder design, our performance evaluation in Section V shows that by itself, it is still lacking.

We have shown that traditional ad hoc routing solutions suffer with frequent topology changes in mobile environments. We also illustrate that traditional wireless sensor routing solutions face routing difficulties in mobile environments due to reliance on local decisions and costly neighbor tables. To address these concerns, we propose the use of Sequential Monte Carlo (SMC) estimation to increase the prediction accuracy of the sink location over multiple hops. We are aware that SMC estimation has been used to improve tracking/localization [36] [37] and sensor fusion [38] performance. These previous studies also point out that SMC is a better choice than other prediction techniques, for it is more accurate than Kalman filters and incurs less overhead than Markov models. We integrate this SMC estimation technique into Sidewinder in a low-overhead manner to account for the computational and bandwidth restrictions present in wireless sensor networks.

III. SIDEWINDER OVERVIEW

Our Sidewinder protocol lies between the transport and MAC layers, as shown in Figure 3. It gets data from the transport layer and forwards it to sink nodes. It also gets global time and individual node location information from the time synchronization and localization protocols, which have been extensively studied in wireless sensor networks. Any existing MAC protocols, like B-MAC [39], can be used below Sidewinder. The predictive multi-hop forwarding functionality is achieved through combined efforts of four modules: Sequential Monte Carlo (SMC) Prediction, Limited Flooding, Mobility Monitor and Adaptive Update.

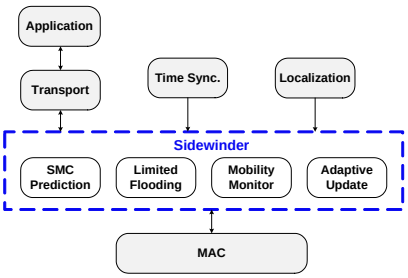


Fig. 3. Sidewinder Architecture

The SMC Prediction module utilizes partial knowledge of sink location from all intermediate nodes in the routing path to make a combined sink location estimate, and hence makes the informed local data forwarding decision dynamically. This holistic prediction of sink location based on distributed knowledge among intermediate nodes facilitates in-situ data forwarding to a mobile sink. This module enables Sidewinder to achieve high packet delivery ratio and low time delay in mobile sensor network environments.

While the SMC Prediction module enables Sidewinder to forward data close to a mobile sink, the Limited Flooding module ensures that data finally reaches the sink. When a forwarding node is a one-hop neighbor of the sink node, data packets received are passed to the Limited Flooding module, which broadcasts them to all the node's two-hop neighbors, ensuring the data reaches the sink even if the sink has moved out of range of the forwarder.

Since group mobility is a common characteristic of buoyant sensor networks or mobile sensor networks such as ZebraNet [6] it is important to measure individual nodes' mobility as well as group mobility. Mobility can be measued based on nodes' locations, which are obtained from localization protocols. The Mobility Monitor module serves this purpose, providing the measured ground truth for SMC Prediction. The Adaptive Update module provides a link between the Mobility Monitor and SMC Prediction, disseminating sink location and mobility information to the network. The Adaptive Update module updates the network in a time and space adaptive manner to save energy and reduce communication overhead.

We now explain in detail SMC Prediction, Mobility Monitor, and Adaptive Update. The Limited Flooding module is comparatively simple and hence not elaborated.

IV. Detailed Sidewinder Design

In this section, we first discuss the general idea of how the Sequential Monte Carlo theory can be integrated into multi-hop data forwarding. Then, we discuss how this design can be trimmed to fit into sensor network environments, where bandwidth and energy are limited. Finally, we present details of each phase in SMC Prediction-based data forwarding along with adaptive sink update.

A. SMC Prediction Concept

During multi-hop data forwarding toward a mobile sink, the goal of the SMC Prediction module is to determine the current sink location using possible locations estimated by previous hops as well as the current hop. It consists of four phases: initialization, prediction, filtering, and resampling.

In initialization, N possible sink locations are generated by a source node based on the last heard sink location, group and random velocities. These N locations form a possible location distribution of the mobile sink. In the prediction phase, a forwarding node uses both the N sink locations generated by the previous node as well as its own knowledge of sink location, group and random velocities to predict current sink locations. The filtering phase allows a forwarding node to use both the previous node's and its own sink location predictions to eliminate impossible sink locations. Finally, in the resampling phase, a forwarding node uses its own sink location information to generate new possible sink locations to replace those eliminated in the filtering phase. In this way, a data packet can be forwarded towards a continuously corrected, predicted sink location, using the knowledge distributed on the nodes in the path.

Several papers [28] [29] [30] [31] show that to deal with frequent topology changes, it is better to let all nodes that overhear the forwarded data to compete for the next hop forwarding, rather than assigning a specific node among a frequently updated neighbor table to forward the data. We incorporate this zone-based forwarding wisdom in Sidewinder: if a node hears a data forwarding packet and also finds itself currently within the 60° forwarding zone facing the sink, it competes for the next hop forwarding; Otherwise, it ignores the data packet. A 60° forwarding area has been shown in [29] to reduce redundant forwarding, so we integrate this forwarding area size into Sidewinder. When a node competes to forward to the next hop, it sets a backoff timer with respect to the forward progress to the estimated sink area center. A competitor within this 60° zone overhears this data forwarding and cancels its backoff timer.

B. SMC adapted to Wireless Sensor Networks

Since communication is far more expensive than computation in sensor networks, it is necessary to minimize the communication overhead of integrating SMC Prediction into multi-hop routing. The main overhead is to send the N estimated sink locations to the next hop. If 4 bytes are used to represent a node location (x,y), $4N$ bytes are needed to represent N locations. Assuming $N = 8$ for achieving reasonable prediction accuracy, 32 bytes are needed for each data forwarding decision. Considering that a typical sensor network packet size is ~ 40 bytes, this is too much overhead, and hence alternative representations are needed.

One alternative is to consider a Convex Hull algorithm like Quickhull [40]. Instead of using all N locations, QuickHull uses Q edges to represent the potential sink area. However, Quickhull is not able to represent the real location distribution of the sink within the potential Convex Hull area, and hence reduces the prediction accuracy.

Another alternative is to consider clustering techniques, such as QT [41], that can group N predicted sink locations

into Q clusters. To represent each cluster, 4 bytes are needed for the center location, 1 byte is needed for the radius, and 1 byte is needed for the number of locations in this cluster. So representing Q clusters needs $6Q$ bytes. If $Q = 3 \sim 4$, then 18∼24 bytes are needed to forward a data packet in each hop, which is also too much overhead. So, a conventional clustering technique is also not a good choice.

Therefore, we propose a sector method that uses a one-dimension, rather than the conventional two-dimension, clustering technique to reduce communication and energy overhead while preserving the sink location distribution in the predicted area. Although reducing clustering from two dimensions to one loses some information, it does not produce a noticeable performance impact. This is because a multi-hop data forwarding decision only needs to know the orientation and distribution of the predicted sink locations from the forwarding node. The distances from the forwarding node to these possible locations are unnecessary. Therefore, in our sector method, a 60° forwarding area focused on the estimated sink area center is used and is also split into Q sectors, with the number of possible sink locations inside each sector calculated to represent the sink location distribution in the forwarding direction. It is effective to just transmit the number of possible sink locations in each sector, rather than all possible sink locations, so the total cost is reduced to just Q bytes.

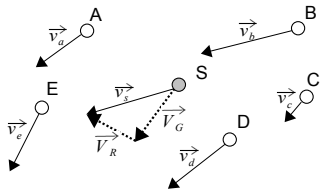


Fig. 4. A sink node S computes its group, random, and neighbor velocities.

C. Mobility Monitor

To understand the SMC forwarding approach and its relationship with node mobility, we first describe the calculation of group and random velocities. We determine the general movement of network as a whole as well (group velocity) as individual nodes' random deviations from that group movement (random velocity). Since there exists an extensive study of retrieving accurate sensor locations through GPS or localization algorithms, in Sidewinder, we assume that each node can get its own location from the localization module as denoted in Figure 3. Based on location history, each node can compute its own velocity. A sink node can also obtain its neighbors' velocities through local beaconing, and combine them with its own velocity $\overrightarrow{v_s}$ to compute its group and random velocities. As shown in Figure 4, when sink S obtains its own velocity $\overrightarrow{v_s}$ and its neighbors' velocities $\overrightarrow{v_a}$, $\overrightarrow{v_b}$, $\overrightarrow{v_c}$, $\overrightarrow{v_d}$, and $\overrightarrow{v_e}$, its group velocity can be computed as the averaged velocity: $\overrightarrow{V_G} = (\overrightarrow{v_a} + \overrightarrow{v_b} + \overrightarrow{v_c} + \overrightarrow{v_d} + \overrightarrow{v_e} + \overrightarrow{v_s})/6$, and the its random velocity $\overrightarrow{V_R}$ can be computed as the difference between absolute and group velocities: $\overrightarrow{V_R} = \overrightarrow{v_s} - \overrightarrow{V_G}$. With this concept in mind, we now explain our SMC approach.

D. SMC Initialization

In initialization, a source node predicts sink locations. It also forwards the predicted locations together with application data towards a mobile sink. The source node's neighbors overhear the forwarded data, and those currently lying in the specified 60° forwarding zone compete for the next hop forwarding.

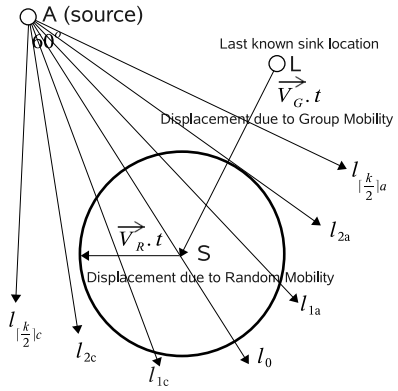


Fig. 5. A source node constructs an estimated sink area based on the last known location, group and random velocities of a sink node.

As shown in Figure 5, a source node (A) computes an estimated sink area, based on the stored sink location (S), group velocity ($\overrightarrow{V_G}$), and random velocity ($\overrightarrow{V_R}$), received through the last sink update, which will be discussed in Section IV-F. The group velocity ($\overrightarrow{V_G}$) is multiplied by the time (t) elapsed since the time that A receives the last adaptive update. The displacement due to group mobility ($\overrightarrow{V_G} \cdot t$), combined with the last known location (L) of the sink node yields the center (S) of the estimated sink area. The random velocity ($\overrightarrow{V_R}$) of the sink node multiplied by the time (t) elapsed since the last update yields the sink random mobility ($\overrightarrow{V_R} \cdot t$), or the radius of the estimated sink area.

After computing the estimated sink area, the source node constructs a 60° forwarding area focused on the estimated sink area center, and also splits it into $1 \leq Q \leq 8$ sectors with a central angle of $\frac{60}{Q}$, as shown in Figure 5. Q is specified at runtime and it is determined through evaluation that more than 8 sectors results in significant communication overhead in comparison with any accuracy gains. Each sector has an inner boundary, $l_i c$ (clockwise) or $l_i a$ (counter-clockwise), and outer boundary, $l_{i+1} c$ or $l_{i+1} a$, with respect to the source-estimated sink area center line l_0.

As shown in Figure 6, the slopes of sector boundaries are first computed from l_0 to $l_{\lceil \frac{Q}{2} \rceil c}$ sequentially in a clockwise manner, and then from l_0 to $l_{\lceil \frac{Q}{2} \rceil a}$ sequentially in a counter-clockwise manner.

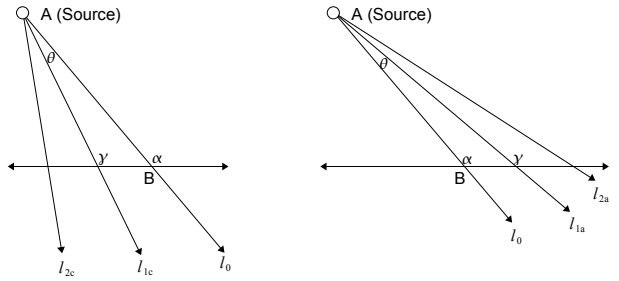
Denoting locations of the source and estimated sink center as (x_A, y_A) and (x_B, y_B), respectively, the slope of l_0 (or line AB) can be computed as:

$$\tan(\alpha) = \frac{|y_A - y_B|}{|x_A - x_B|} \tag{1}$$

The slope of l_0 is then used in combination with the sector central angle θ to compute $tan(\gamma)$, the slope of l_{1c}. Since $1 \leq Q \leq 8$, and hence there is a fixed, small number of possible θ values, the value of $\tan(\theta)$ can be obtained from a table lookup, rather than using the limited computation resources of small sensor devices.

$$tan(\gamma) = \tan(\alpha - \theta) = \frac{\tan(\alpha) - \tan(\theta)}{1 + \tan(\alpha)\tan(\theta)} \quad (2)$$

Using Equation 2, for a given sector boundary l_{ic} with known slope $\tan(\alpha)$, the slope for the adjacent sector boundary l_{i+1c} can be calculated as $\tan(\gamma)$.



(a) Calculating the slope of clockwise boundaries (b) Calculating the slope of counter-clockwise boundaries

Fig. 6. The slope of a boundary l_{1j}, where $j \in \{c, a\}$, can be calculated using the known values $\tan(\theta)$ and slope of l_0, $\tan(\alpha)$. The slope of l_{2j} can then be calculated from the slope of l_{1j}.

Once all the sector boundaries from l_{1c} to $l_{\lceil \frac{Q}{2} \rceil c}$ have been computed, the sector boundaries from l_{1a} to $l_{\lceil \frac{Q}{2} \rceil a}$ are computed sequentially in a counter-clockwise fashion. The process mirrors that used for sector boundaries l_{1c} to $l_{\lceil \frac{Q}{2} \rceil c}$. For example, using Equation 1 and the sector central angle θ, the slope of l_{1a}, or $\tan(\gamma)$ can be calculated as:

$$tan(\gamma) = tan(\alpha + \theta) = \frac{\tan(\alpha) + \tan(\theta)}{1 - \tan(\alpha)\tan(\theta)} \quad (3)$$

With the boundaries determined for each sector, the source node then generates N random locations, $8 \leq N \leq 64$, uniformly within the estimated sink area (dashed circle in Figure 7), to represent possible sink locations (white points). As with the number of sectors, it is determined during evaluation that using more than 64 locations incurs a great amount of computational overhead in comparison to a small increase in forwarding accuracy. For each generated location R, we compute which sector in which it is located by comparing the slope of the line that connects this location and the source A (line AR) with that of sector boundaries. With the number of possible sink locations in each sector computed, a source node piggybacks this information in a data packet, together with locations of the source and estimated sink center. Compared with existing geographic forwarding-based protocol like GPSR [10], only Q extra bytes are needed for forwarding to a mobile sink. In Algorithm 1, we summarize the initialization phase.

Algorithm 1 Initialization

Input: The number of possible sink locations to predict N, the source location $A(x_A, y_A)$, the estimated sink center $S(x_S, y_S)$, and radius r_S ($r_S = V_R \cdot t$ as shown in Figure 5).
Output: The number of possible sink locations in each sector, C_i, $1 \leq i \leq Q$

```
cntLocations = 0;
while (cntLocations < N) do
    Generate random location R(x_R, y_R) inside circle (S, r_S);
    if (R is in sector i) then
        C_i++;
        cntLocations++;
    end if
end while
```

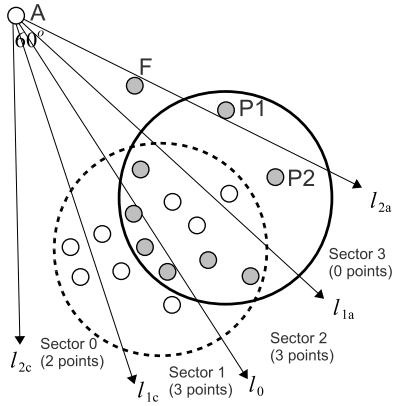


Fig. 7. The source (A), generates 8 possible sink locations (white) in its estimated sink area (dashed circle) and determines the number of points in each sector before transmitting. Node F, the forwarder, creates a new estimated sink area (solid circle) and node A's sectors. F generates new possible sink locations (gray) in overlapping areas of A's and its own predictions, based on the number of points in A's corresponding sectors. Remaining points (P1 and P2) are created uniformly within the estimated sink area until there are N possible sink locations.

E. SMC Prediction, Filtering, and Resampling

After a data packet is forwarded from a source to the next hop, three things happen: 1) Prediction: the N sink locations generated by the source is used to predict current sink locations again[1]. Also, the current node predicts sink locations based on its own knowledge; 2) Filtering: the current node makes use of both the previous node's and its own sink location predictions to filter out impossible sink locations; 3) Resampling: the current node uses its own knowledge to generate new possible sink locations to replace those eliminated in the filtering phase.

Since the three phases are closely related, we explain them together. As shown in Figure 7, when node F receives a data packet from node A, several items are reconstructed. From the information piggybacked in the data packet, node F determines the following: the 60° forwarding zone from

[1]The elapsed time between when the previous node predicts the sink location and when the current node needs to predict the sink location is the same as a single hop data forwarding time. It is usually a few milliseconds. Since the elapsed time is so small, the sink movement within such a period can be ignored. Therefore, an intermediate forwarding node can just use the N possible sink locations generated by a previous node to reconstruct, rather than predict, the current sink location.

node A, the 4 sectors in the forwarding zone, and also the 2/3/3/0 sink location distribution within the sectors. Node F also predicts possible sink locations (gray points), based on its own knowledge of the sink location as well as its group and random velocities received in the last sink update.

In the filtering process, the accumulated sink prediction, which is represented by the number of locations in each sector, is filtered using node F's prediction. The accumulated sink predictions are plotted within the solid circle in Figure 7. For instance, since sector 1 and 2 overlap with node F's prediction circle, the 3/3 location distribution is regenerated in the corresponding overlapping area between sectors and the solid circle. Also, since sector 0 has no overlapping with the circle predicted by node F, the 2 locations in this sector are filtered out. Since this filtering phase only regenerates 6 sink locations, 2 more locations are needed to form the new prediction. As shown in Figure 7, these new locations P1 and P2 are resampled uniformly from node F's prediction.

With a similar technique as presented in subsection IV-D, 4 sectors can be formed at F towards the newly estimated sink center. Then, the location of F, the estimated sink center of F, and the number of locations in each sector are piggybacked again in the data packet and forwarded to the next hop.

The core of the filtering and resampling phases is summarized in Algorithm 2. One important aspect of this algorithm is that we do not directly generate the specified number (C_i) of locations within the overlapping area of each sector and the new estimation area (S, r_S), since that requires too much computation for sensor nodes. Instead, we generate enough random locations (10 times the expected number N) within the circular area, and determine the sector in which they are located. We are aware that if an overlapping area of a sector and the new estimation area is very small, it may not get any location generated inside, even though its C_i value is not zero. This does not have an noticeable impact on accuracy since only a tiny overlapping area is ignored when the total number of random locations is 10 times that of what is expected.

F. Adaptive Update

The SMC Prediction module is updated with sink location, group and random mobilities in a time and space adaptive manner via the Adaptive Update module. The method used is similar to that in [19], where nodes notify each other of their location at a rate based on distance and relative velocity in order to reduce bandwidth and energy costs. In Sidewinder, a sink node updates its location and mobility behavior according to Equation 4 so that all Limited Flooding initiated two hops away will be successful.

$$\mid \overrightarrow{V_R} \mid \times t \geq 2r \tag{4}$$

In Equation 4, $\overrightarrow{V_R}$ refers to the sink random velocity, t is the time since the last sink update, and r is the radio range, which is empirically configured. With this time-adaptive update, all one-hop neighbors of the sink at its last update will be no more than two hops away from the sink before the sink updates again, allowing all Limited Flooding packets to reach the sink.

Algorithm 2 Filtering and Resampling

Input: The number of possible sink locations to predict N, the previous node location $A(x_A, y_A)$, the estimated sink center $S(x_S, y_S)$ and radius r_S ($r_S = V_R \cdot t$ as shown in Figure 5) of the current node F, the number of possible sink locations in sector i of node A, C_i, $1 \leq i \leq Q$.
Output: Newly predicted N sink locations for node F.

```
cntLocations = 0; cntSectorNodes = 0;
{Filtering}
while (cntSectorNodes < N * 10) do
  Generate random location R(x_R, y_R) inside circle (S, r_S);
  if (R is in sector i of node A) then
    C_i- -;
    cntLocations++;
    If C_i ≥ 0, mark R as an estimated sink location;
  end if
  cntSectorNodes++;
end while
{Resampling}
while (cntLocations < N) do
  Generate random location R(x_R, y_R) inside circle (S, r_S);
  Mark R as an estimated sink location;
  cntLocations++;
end while
```

Bandwidth and energy are further reduced by decreasing the sink update frequency as distance to the sink increases. To achieve this, each node has a probability P for forwarding a sink update packet. The value of P for a given node is based on its number of hops from the sink, h. If a node is h hops away from the sink, the probability for it to forward a sink update packet, P_h, is computed as:

$$P_h = \alpha^{h-1}, (0 < \alpha \leq 1) \tag{5}$$

In Equation 5, α is set as 1 for the first update, so that all nodes in the network have initial knowledge of sink location and movement behavior. After system initialization, α is reduced to an empirical value (0.2 is used in our evaluation) so that nodes farther away from the sink receive fewer updates. Nodes farthest away from the sink need only a general idea of the sink location and movement behavior; nodes that forward the data have a more accurate picture of the sink location and movement. As data makes its way from the network edge to the sink, more accurate sink information is used at each hop to precisely route data to the sink.

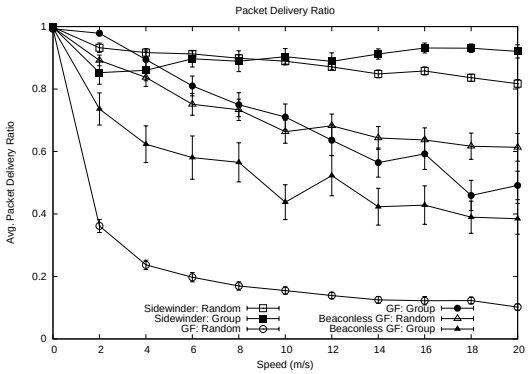
V. Performance Evaluation

Sidewinder is implemented in TinyOS-2.x [42] with nesC [20] and evaluated in TOSSIM [21] using B-MAC [39]. Two mobility models are used in evaluation: Random Waypoint without pause time [33], and the Reference Point Group [43] mobility model. Though more advanced mobility models exist [44], we choose Random Waypoint and Reference Point Group since they are simple and apply to a large number of possible scenarios, ranging from flood tracking to movements of search and rescue teams [32]. When Random Waypoint mobility is used, 500 nodes are uniformly deployed in an area of 215m x 215m, with the radio range of 25m. When

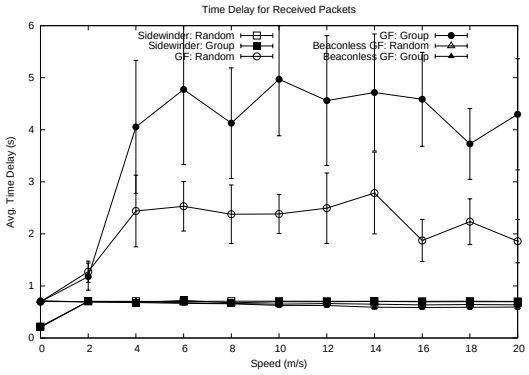
Reference Point Group mobility is used, the deployment region is increased to 2000m x 2000m to allow for group movement. Also for Reference Point Group mobility, the group radius is set to 120m to maintain the same node density as Random Waypoint and each node's random movement is set to the maximum group speed. In both mobility cases, 3 sources are randomly chosen to report sampled readings to the same sink, at the frequency of one packet per second. In Sidewinder, the Mobility Monitor module beacons location information every 10s and the maximum backoff window in zone-based forwarding competition is 128ms. In GF, each node beacons every 1.5s and a neighbor table entry expires every 6.7s, which is identical to the evaluation of GF in [10]. We also replace the neighbor table maintenance in GF with a 60° zone-based forwarding strategy [28] [29] [30] [31], but without SMC prediction, and call the modified protocol Beaconless GF. In GF and Beaconless GF, a sink node floods its location every 10s. We repeat each evaluation 100 times and present the averaged results in Figure 8, together with 90% confidence intervals. As shown in Figure 8 (a), Sidewinder significantly outperforms GF and Beaconless GF when nodes are mobile. For instance, when node speed is 20m/s, Sidewinder achieves 92% packet delivery ratio in group mobility, which is 52% higher than that of Beaconless GF and 42% higher than that of GF. Sidewinder also exhibits an 82% packet delivery ratio in random mobility, which is 20% higher than that of Beaconless GF and 72% than that of GF. This is because Sidewinder's SMC prediction continuously corrects the data forwarding direction towards the mobile sink and the zone-based forwarding tolerates excessive topology changes due to mobility. GF does not have any of these mobility-aiding techniques and Beaconless GF does not have the SMC prediction technique. For the same reasons, Sidewinder always maintains a high packet delivery ratio, ≥ 80, while GF and Beaconless GF suffer from increasing node mobility.

In Figure 8 (a), we also observe that Beaconless GF achieves a higher packet delivery ratio than GF in random but not group mobility. Beaconless GF achieves 50% higher performance than GF in random mobility, because GF's neighbor table maintenance is negatively impacted by excessive topology changes. These topology changes are tolerated by the zone-based forwarding in Beaconless GF. In group mobility, GF achieves 10% higher performance than Beaconless GF. This is because the relative movement between nodes is small in comparison with a random mobility model with the same average node velocity. However, with group mobility, the 60° forwarding area constraint in Beaconless GF eliminates possible routing paths that GF can find. This also explains why Sidewinder achieves less than a 100% packet delivery ratio when there is mobility. We plan to address this issue in future, e.g., by borrowing the wisdom of face routing [10].

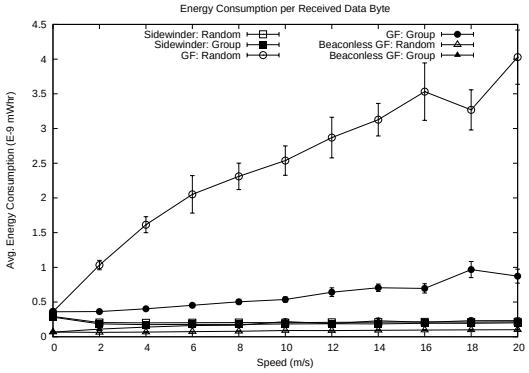
We also measure the end-to-end time delay and energy consumption per successfully delivered data byte to the sink, and present results in Figure 8 (b) and (c). Figure 8 (b) demonstrates that Sidewinder and Beaconless GF achieve similar but much lower time delay than GF, especially in random mobility.



(a) End-to-End Packet Delivery Ratio



(b) End-to-End Time Delay



(c) Energy Use per Delivered Data Byte

Fig. 8. Performance Evaluation

This is due to routing loops in GF. Such loops are caused by the comparative random movement between neighboring nodes, and that also explains why the measured time delay in the random mobility case is significantly higher than that in the group mobility case. Since energy conservation is of utmost concern in wireless sensor networks, we depict Figure 8 (c) as the amount of overhead used by Sidewinder, Beaconless GF, and GF. Increases in number of radio transmissions, packet sizes, and computation will result in increased energy consumption. Figure 8 (c) demonstrates that Sidewinder and Beaconless GF consume similar but much less energy than GF, which is due to two reasons. First, GF expends significant energy on high frequency beaconing for updating neighbor

tables. Second, GF wastes more energy on packets that fail to be delivered to the mobile sink. GF also demonstrates better energy efficiency in group than random mobility, since multi-hop routing failures are more prevalent with the increased number of topology changes in random mobility.

VI. Conclusions and Future Work

This paper presents Sidewinder, a novel protocol for in-situ data collection in mobile wireless sensor networks. We show through quantitative evaluation that traditional ad hoc and wireless sensor routing solutions fail in highly mobile environments. Thus, Sidewinder addresses the issues of highly dynamic network topologies and static route failures with Sequential Monte Carlo prediction of sink locations. As a data packet makes its way from source to sink, the sink location prediction computed at each node is combined and updated with each successive hop, increasing prediction accuracy. We integrate this forwarding mechanism into Sidewinder using a one-dimensional clustering technique that preserves sink location prediction accuracy while minimizing bandwidth and energy overhead. Our performance evaluation in TOSSIM demonstrates that Sidewinder significantly outperforms state-of-the-art solutions in packet delivery ratio, time delay, and energy efficiency.

References

[1] J. Allred, A. B. Hasan, S. Panichsakul, W. Pisano, P. Gray, and J. Huang, "SensorFlock: An Airborne Wireless Sensor Network of Micro-Air Vehicles," in *ACM SenSys*, 2007.

[2] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G-S. Ahn, and A. T. Campbell, "The BikeNet Mobile Sensing System for Cyclist Experience Mapping," in *ACM SenSys*, 2007.

[3] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring," in *ACM MobiSys*, 2008.

[4] B. Hull, V. Bychkovsky, K. Chen, M. Goraczko, E. Shih, Y. Zhang, et al., "Cartel: A Distributed Mobile Sensor Computing System," in *ACM SenSys*, 2006.

[5] T. Wark, C. Crossman, W. Hu, Y. Guo, P. Valencia, P. Sikka, et al., "The Design and Evaluation of a Mobile Sensor/Actuator Network for Automous Animal Control," in *ACM/IEEE IPSN*, 2007.

[6] T. Liu, C. M. Sadler, P. Zhang, and M. R. Martonosi, "Implementing Software on Resource-Constrained Mobile Sensors: Experimences with Impala and ZebraNet," in *ACM MobiSys*, 2004.

[7] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "Data Collection, Storage, and Retrieval with an Underwater Sensor Network," in *ACM SenSys*, 2005.

[8] J. Zhao and R. Govindan, "Understanding Packet Delivery Performance in Dense Wireless Sensor Networks," in *ACM SenSys*, 2003.

[9] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," in *ACM SenSys*, 2003.

[10] B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *ACM MobiCom*, 2000.

[11] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker, "Geographic Routing Made Practical," *USENIX NSDI*, 2005.

[12] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamachari, "EnergyEfficient Forwarding Strategies for Geographic Routing in Lossy Wireless Sensor Networks," in *ACM SenSys*, 2004.

[13] T. He, J. A. Stankovic, C. Lu, and T. F. Abdelzaher, "SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks," in *IEEE ICDCS*, 2003.

[14] S. Funke and N. Milosavljevic, "Guaranteed-delivery Geographic Routing under Uncertain Node Locations," in *IEEE INFOCOM*, 2007.

[15] C. E. Perkins and E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing," in *IEEE WMCSA*, 1999.

[16] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly Resilient, Energy Efficient Multipath Routing in Wireless Sensor Networks," in *ACM MC2R*, 2002.

[17] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," 1996, In Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Kluwer Academic Publishers, 1996.

[18] Y. B. Ko and N. H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," in *ACM MobiCom*, 1998.

[19] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," in *ACM MobiCom*, 1998.

[20] D. Gay, P. Levis, R. V. Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC Language: A Holistic Approach to Network Embedded Systems," in *ACM PLDI*, 2003.

[21] P.Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications," in *ACM SenSys*, 2004.

[22] "MultiHopLQI," http://www.tinyos.net/tinyos-1.x/tos/lib/MultiHopLQI/.

[23] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," in *ACM MobiCom*, 2000.

[24] D. Son, A. Helmy, and B. Krishnamachari, "The Effect of Mobility-induced Location Errors on Geographic Routing in Mobile Ad Hoc and Sensor Networks: Analysis and Improvement using Mobility Prediction," *IEEE Trans. Mobile Comput.*, 2004.

[25] A. Nguyen, N. Milosavljevic, Q. Fang, J. Gao, and L. J. Guibas, "Landmark Selection and Greedy Landmark-Descent Routing for Sensor Networksk," in *IEEE INFOCOM*, 2007.

[26] A. Rao, S. Ratnasamy amd C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *ACM MobiCom*, 2003.

[27] Q. Cao and T. F. Abdelzaher, "A Scalable Logical Coordinates Framework for Routing in Wireless Sensor Networks," in *IEEE RTSS*, 2004.

[28] S. Biswas and R. Morris, "ExOR: Opportunistic Multi-Hop Routing for Wireless Networks," in *ACM SIGCOMM*, 2005.

[29] M. Heissenbuttel, T. Braun, T. Bernoulli, and M. Walchie, "BLR:Beacon-Less Routing Algorithm for Mobile Ad-Hoc Networks," in *Elsevier Computer Communication*, 2003.

[30] P. Larsson, "Selection diversity forwarding in a multihop packet radio network with fading channel and capture," in *ACM MC2R*, 2001.

[31] M. Zorzi and R. Rao, "Geographic Random Forwarding (GeRaF) for ad hoc and sensor networks: multihop performance," in *IEEE TMC*, 2003.

[32] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Networks Research," in *IWCMC*, 2002.

[33] J. Yoon, M. Liu, and B. Noble, "Sound Mobility Models," in *ACM MobiCom*, 2003.

[34] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Models and Solutions for Radio Irregularity in Wireless Sensor Networks," in *ACM Transactions on Sensor Networks*, 2006.

[35] B. Karp, *Geographic Routing for Wireless Networks*, Ph.D. thesis, Harvard University, Cambridge, MA, 2000.

[36] L. Hu and D. Evans, "Localization for Mobile Sensor Networks," in *ACM MobiCom*, 2004.

[37] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation," in *IEE Proceedings*, 1993.

[38] D. Guo and X. Wang, "Dynamic sensor collaboration via sequential Monte Carlo," in *IEEE JSAC*, 2004.

[39] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *ACM SenSys*, 2004.

[40] C. B. Barber, D. Dobkin, and H. Huhdanpaa, "The Quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, 1996.

[41] L. Heyer, S. Kruglyak, and S. Yooseph, "Exploring Expression Data: Identification and Analysis of Coexpressed Genes," *Genome Research*, 1999.

[42] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System Architecture Directions for Networked Sensors," in *ASPLOS-IX*, 2000.

[43] X. Hong anf M. Gerla, G. Pei, and C. Chang, "A Group Mobility Model for Ad Hoc Wireless Networks," in *ACM MSWiM*, 1999.

[44] Z. Zaidi, B. Mark, and R. Thomas, "A Two-tier Representation of Node Mobility in Ad Hoc Networks," in *IEEE SECON*, 2004.